MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1231 May 1990

Repairing Learned Knowledge Using Experience

Patrick H. Winston and Satyajit Rao

Abstract

Explanation-based learning occurs when something useful is retained from an explanation exercise, usually an explanation of how some particular problem can be solved. If explanation is based on sound theory, then the learning process speeds up future problem solving, but the scope of the learning-augmented theory remains unchanged. In contrast, if explanation is based on fragmentary and often defective experience, explanation can be a guide to when that experience can be deployed. Thus one kind of explanation provides speed up; another kind of explanation provides new knowledge. Experience is not sound theory, however, and wrong things may be learned accidentally, as subsequent failures will likely demonstrate. In this paper, we describe ways to isolate the facts that cause failures, ways to explain why those facts cause problems, and ways to repair learning mistakes. In particular, our program learns to distinguish pails from cups after learned knowledge about cups leads a recognition program to think pails are cups.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defence under Office of Naval Research contract N00014-85-K-0124.

Becoming an Expert Requires Repair as well as Acquisition

For most of us, in most domains, we get by with simple rulelike knowledge, handling ordinary situations well enough. But if we choose to become an expert in some domain, we begin to see the unusual cases, which we handle by way of an exception-recognizing mechanism or by repairing our original, rulelike knowledge.

In this paper, we show how to repair rulelike knowledge by exploiting situations where the rulelike knowledge leads to incorrect conclusions. Our implemented program learns to distinguish pails from cups after a learned cup recognition rule incorrectly identifies certain pails as cups.

Our work builds on the vast literature on explanation-based learning (see Mitchell *et al.* [1986] for a review). In particular, we build on the ideas in Winston *et al.* [1983], which explained how to acquire rulelike knowledge about what cups look like in the course of explaining how cups achieve their function.

Our Repair Work Builds on Analogy and Explanation

To understand our approach, it is helpful to review the explanation-based analogical reasoning that produces the original rulelike knowledge.

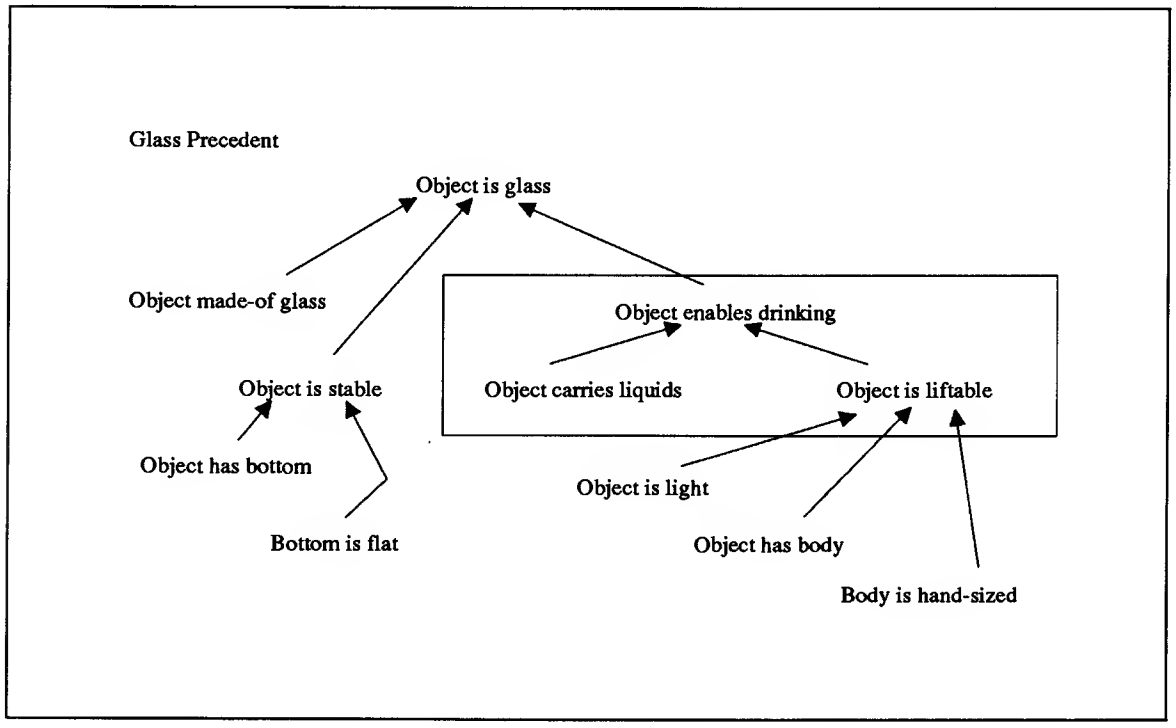

Figure 1. The glass precedent. Precedents include relations that are tied together with *cause* relations, all of which constitute an implied AND tree. The boxed part of the glass precedent helps to bridge a gap in a problem situation shown in the next figure.

First, note that the *cause* relations in precedents constitutes an implicit AND tree with each node in each AND tree corresponding to a relation and each branch corresponding to a relation-linking *cause* relation. Figure 1 shows the *cause* relations in the precedent that describes what it takes to be a glass.

An implicit AND tree can help to bridge the gap in a problem between something that is to be explained and things that are already known. In figure 2 the problem is to show that an unknown object is a cup by connecting the relation, *object is cup?*, to some of the other relations using implicit AND tree fragments from various precedents. In figure 3, the relation, *object is cup*, is connected to some of the other relations using fragments from the brick, glass, bowl, and briefcase precedents along with a functional cup definition expressed itself in the same form as the precedents.

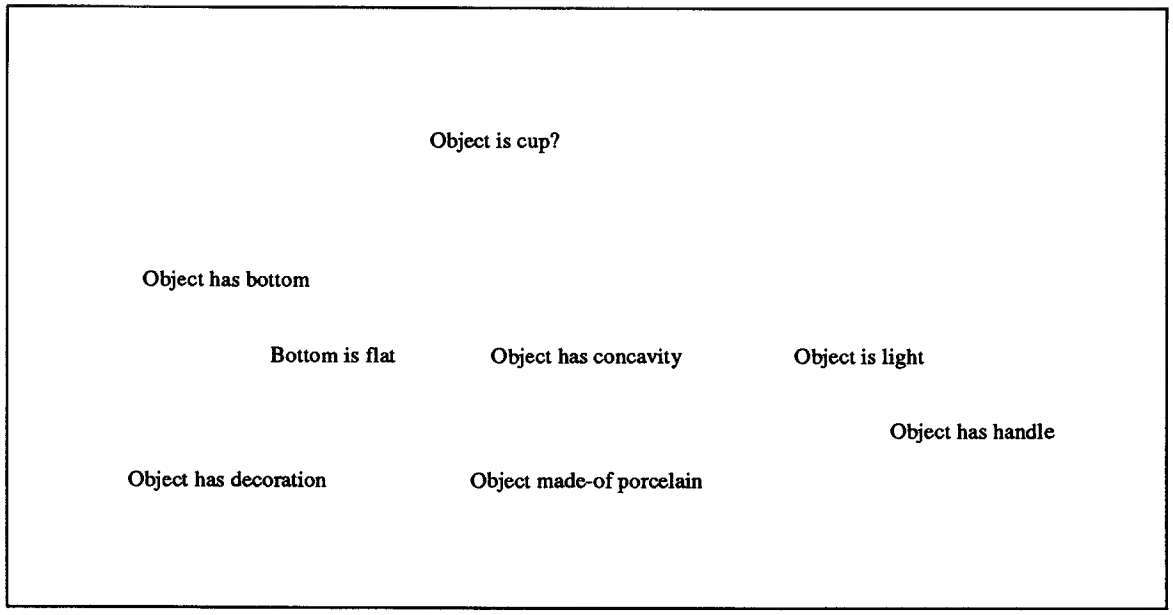

Figure 2. A problem. It contains a relation to be explained, shown with a question mark, and some relations that are known.

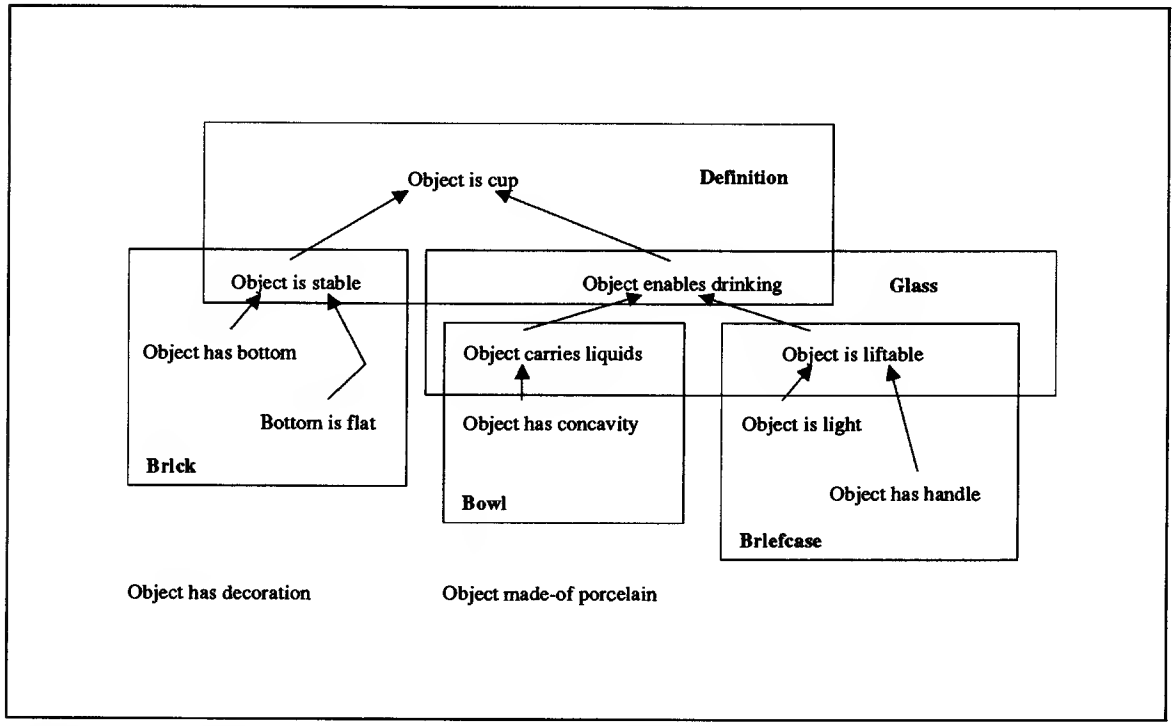

Figure 3. Boxes indicate the origins of the various precedent-supplied AND tree fragments that bridge the gap between the relation to be explained and some of the relations that are known. Only the relevant parts of the precedents are shown.

The gap-filling AND tree, consisting of fragments from one or more definitions and precedents, can be retained for future use. Moreover, the leaves of the AND tree in the new chunk of knowledge can be viewed as the *if* conditions in an implicit *if-then* rule. The root of the AND tree can be viewed as the *then* condition. The other nodes can be viewed as *providing* conditions. The *providing* conditions, like the

terminal *then* condition, are explained by the *if* conditions.

Here the cup-recognition recollection is viewed as an *if-then* rule:

Cup-recognition Recollection:

If	The object has a bottom
	The bottom is flat
	An object has a concavity
	The object is light weight
	The object has a handle
Then	The object is a cup
Providing	The object is stable
	The object enables drinking
	The object carries liquids
	The object is liftable

Note that although the retained knowledge can be viewed as an *if-then* rule, it is really a full-blown, gap-filling AND tree. We call these retained chunks of AND tree *recollections* to emphasize both that something has been remembered and that there has been an explicit re-collection of knowledge that was formerly distributed implicitly across one or more precedents.

Because we choose to express recollections in the same representation as ordinary situation descriptions and definitions, our analogical reasoning machinery does not care whether the source of a gap-bridging AND tree is a precedent, definition, or an already-learned recollection.

Censors shut off conclusions

Finally, because precedents and precedent combinations involve only plausible explanation, there has to be a way of shutting off conclusions. If there is strong, obvious evidence that indicates one of the providing conditions cannot be true in a particular situation, then the explanation that links the *if* conditions to the *then* condition collapses in that particular situation.

A recollection or precedent is called a *censor* whenever it provides evidence that a providing condition in another recollection or precedent cannot be true. Thus censors are just ordinary recollections or precedents used in a special way.

Explanation-based learning offers more than speedup

Importantly, explanation-based learning based on precedents is particularly useful when the learner has weak or faulty knowledge of how individual precedents can be combined into larger explanations. The reason is that particular problems, often supplied by a knowledgeable teacher, provide heuristic evidence about which precedents can be stuck together usefully. Thus the learning is more than just a speed-up phenomenon.

Cups and Pails Illustrate the Problem

The cup-recognition recollection introduced in the previous section can recognize a variety of cups, including porcelain cups and metal cups. Unfortunately, it also recognizes a variety of pails, including metal pails and wooden pails.

Our general approach to improving the situation involves three desiderata. First, the old recollection should be repaired, rather than a totally new one constructed, on the ground that incremental change is less risky than radical change. Second, the repair procedure should exploit failures on the ground that programs should learn from mistakes. And third, the exploitation procedure should use precedents on the ground that experience is the best guide in the absence of sound theory.

These desiderata lead to the following questions:

- How can a program use failures to *isolate* suspicious relations that should perhaps prevent a recollection from being misapplied?
- How can a program use precedents to *explain* why those now isolated suspicious relations should prevent a recollection from being misapplied?
- How can a program use explanations to *repair* a recollection, preventing further misapplication?

Near-Miss Groups Isolate Suspicious Relations

If a metal pail differs from a porcelain cup only in the position of handle attachment, then we would say that the pail is a near miss. Unfortunately, there are many differences, both relevant and irrelevant: the pail is metal, but the cup is porcelain; the metal pail is gray, but the porcelain cup is white with balloons painted on the side; the metal pail carries oats, but the porcelain cup carries coffee.

The metal pail is a *false-success situation*. Let us assume that our recollection-repair system knows that the metal pail in figure 4c is a false success either because a teacher says so or because an effort to drink from it leads to failure.

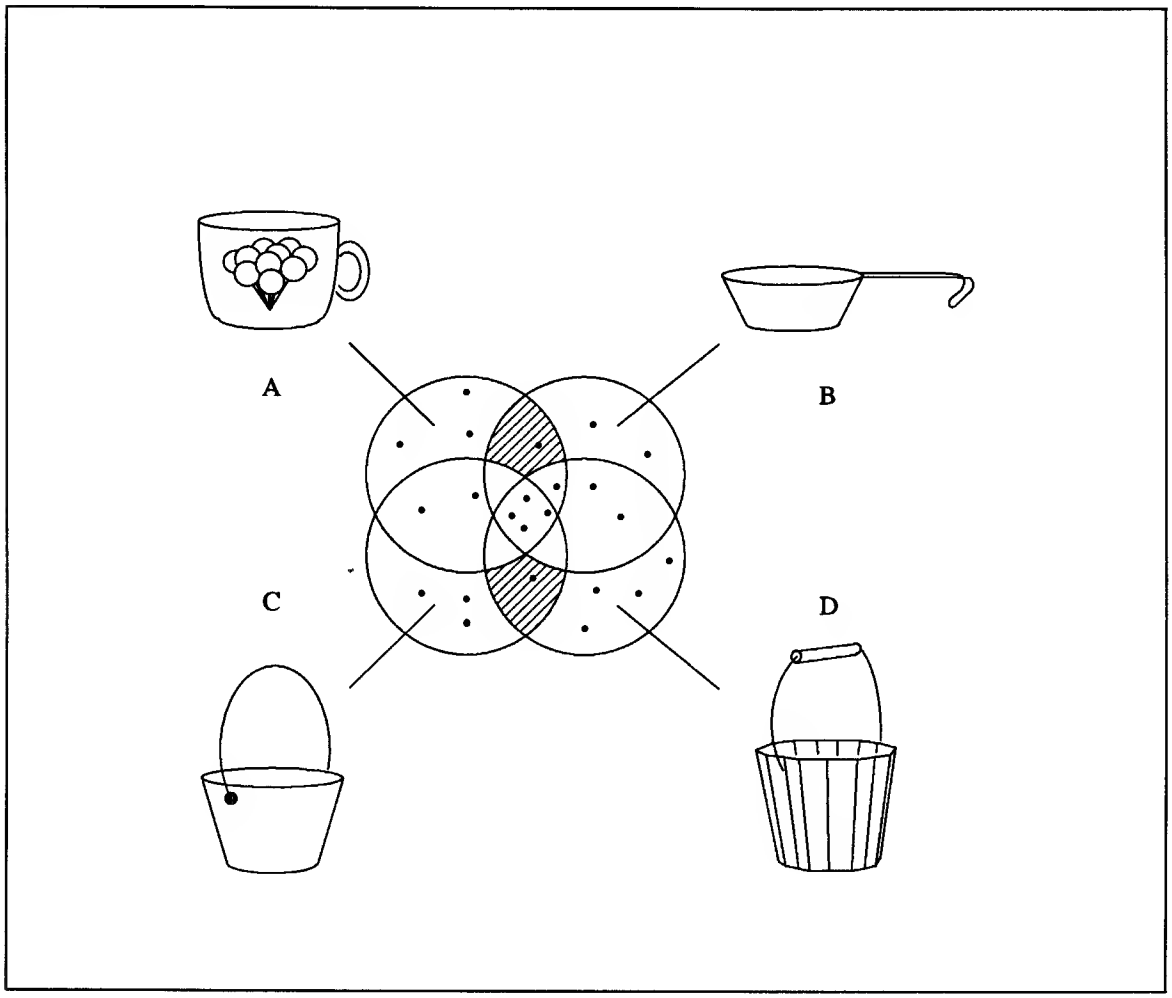

Figure 4. A near-miss group. The dots represent relations. Suspicious relations, the ones in the shaded area, are suspicious because they are in all the true successes but none of the false successes or because they are in all the false successes but none of the true successes.

Let us also assume that our recollection-repair system knows the wooden pail in figure 4d is also a false success. Both pails differ from the porcelain cup in figure 4a in many ways, and similarly, both pails differ from the tin cup in figure 4b in many ways.

Importantly, however, there are fewer ways in which *both* the metal pail and the wooden pail differ from *both* the porcelain cup and the tin cup: for example, the cups have fixed handles, whereas the pails have hinged handles.

As the Venn diagram in figure 4 shows, all four objects can be used together to separate signallike explanations from noiselike distractions. Because the recollection allows all four objects to be recognized as cups, the antecedent relations in the recollection, viewed as an if-then rule, must lie in the intersection of all the relation sets describing those four objects. Similarly, the relations, if any, that distinguish the true-success situations from the false-success situations must lie in the union of the two relation subsets shown shaded in figure 4.

Because the relations in the true-success set and the false-success set are likely candidates for forming explanations, we say that those relations are *suspicious relations*. Also, we say that the situations used to identify the suspicious relations constitute a *near-miss group* because they work together as a group to perform like an single example and a single near miss.

Clearly, the isolation of suspicious relations is just a simple matter of unioning, intersecting, and differencing the relations that appear in the true successes and false successes.

In general, there will be more than one suspicious relation, but the more true successes and false successes we have, the fewer suspicious relations there are likely to be. Note, however, that *isolating* failure-causing relations is not the same as *explaining* failure and *repairing* the flaw.

Suspicious Relation Types Determine Overall Repair Strategy

For suspicious relations that are common to true successes only, there are several possible explanations for how the recollection, viewed as an AND tree, is flawed. For example, a node may involve the wrong relation, a branch corresponding to a particular *cause* relation may be missing, or a chain of two or more branches may have been collapsed into one, eliminating one or more providing conditions that would otherwise be vulnerable to attack by censors. The recollection should be repaired accordingly.

In our example, the relation *handle is fixed* is found in both cups but it is *not* found in either pail. If a program can find a way to include this relation in the rule, then the rule would be more discriminating. It would still recognize the cups, but it would not recognize either pail.

One explanation-free recollection repair would be to include the *handle is fixed* relation in the rule form of the recollection as a new *if* condition:

Repaired Cup-recognition Recollection:

If	The object has a bottom
	The bottom is flat
	An object has a concavity
	The object is light weight
	The object has a handle
	The handle is fixed
Then	The object is a cup
Providing	The object is stable
	The object enables drinking
	The object carries liquids
	The object is liftable

For suspicious relations that are common to false successes only, the failure is the result of not knowing that a providing condition cannot be true. In our example, the relation *handle is hinged* is found in both pails but is *not* found in either cup. If a program can find a way to include this relation in a new censor that attacks one of the providing conditions, that censor would make the rule more discriminating. The rule would still recognize the cups, but it would not recognize either pail.

One explanation-free approach is to manufacture a censor directly out of the *handle is hinged* relation and the *object is not a cup* relation:

New Censor:

If The handle is hinged
Then The object is not a cup.

As described, both the cup repair and the censor creation are *ad hoc* because there is no explanation for why the new *if* condition or the new censor should work. We can do better.

The Solution May Be To Explain the True-Success Suspicious Relations, Changing the Recollection

Before showing how we actually handle true-success suspicious relations, we display the result in our cup-and-pail example. Figure 5 shows both the recollection tree of the original, faulty recollection and the recollection tree of the repaired, correct recollection.

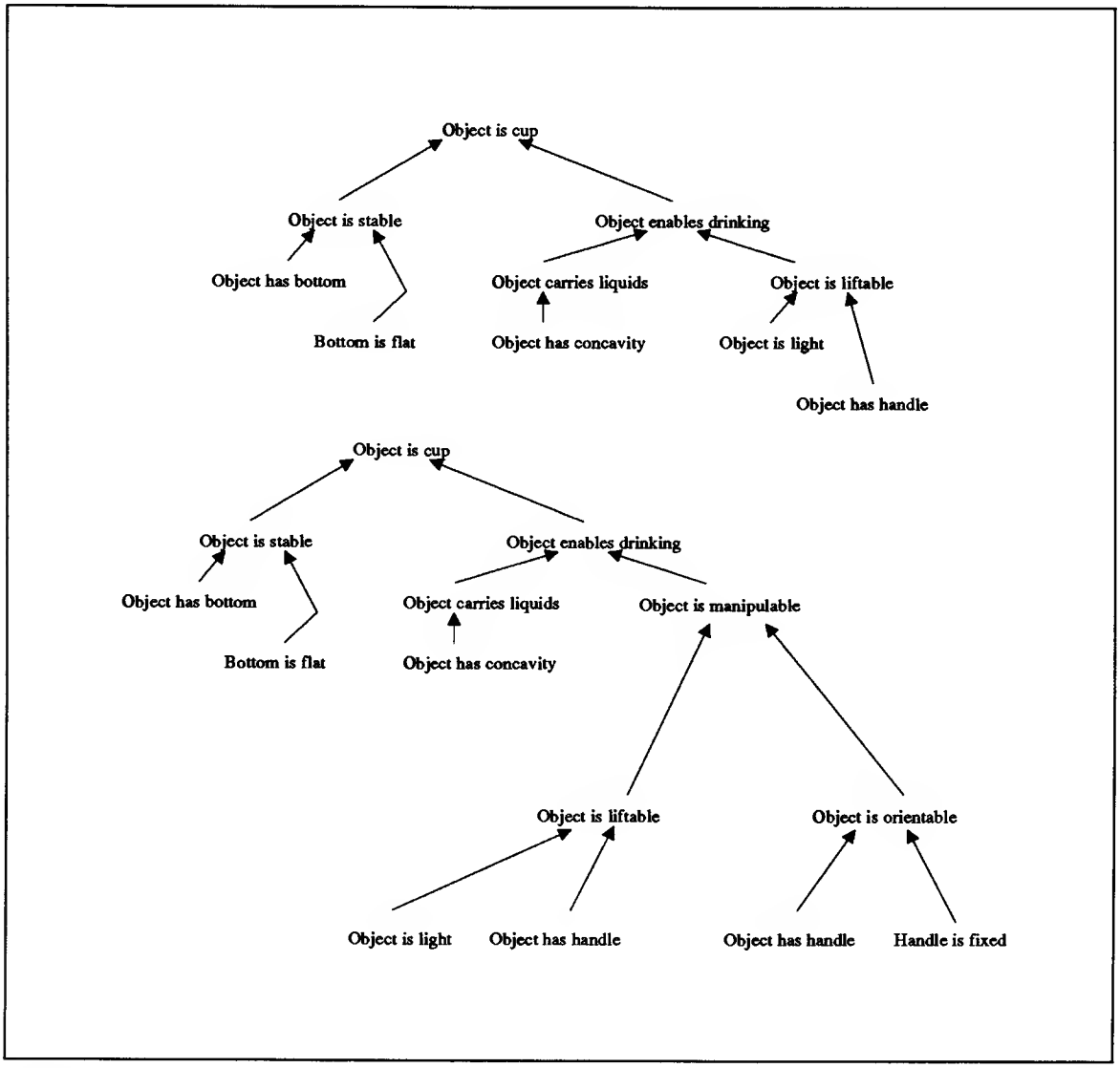

Figure 5. A recollection tree before and after repair. The repaired recollection recognizes only cups; it does not recognize pails.

Comparing the recollection trees of the faulty and repaired recollections, you see that the *handle is fixed* relation, which is common to the true successes, now appears in the repaired recollection tree. There are also two other new relations *object is manipulable* and *object is orientable.*

The old recollection was too general because you cannot be sure you can drink from an object just because it carries liquids and is liftable—it has to be orientable by virtue of having a fixed handle. The new recollection still recognizes the cups, but the increased specificity of the recollection prevents it from recognizing the pails.

To make the repair, our program does a breadth-first reexamination of all the relations in the recollection tree, looking for a relation with an explanation that needs replacement. For each relation reexamined, our program looks for a collection of precedents that ties the reexamined relation to at least one of the true-success suspicious relations along with some or all of the relations that lie in the intersection of all the successes.

If such a collection of precedents is found, it is used to replace the subtree beneath the reexamined relation, thus explaining the reexamined relation in a new way.

The new explanation should be as short as possible because the longer the chain of precedent-supplied *cause* relations, the less reliable the conclusion. After all, the contributing precedents supply *cause* relations that are only likely, not certain. Consequently, we initially limit the reexamination effort to the following precedents:

- The precedents originally used to form the recollection. These are included in the expectation that much of the recollection will be unchanged and therefore constructable from the original precedents. These original precedents constitute the initial *top set.*
- Those precedents in which one of the true-success suspicious relations causes something. These precedents constitute the initial *bottom set.*

When our program reexamines a relation, it looks for a way of explaining that relation using all but one of the precedents in the combined top and bottom sets. The exception is the precedent that contained the cause relations that explained the reexamined relation. This precedent is omitted so as to explore the hypothesis that it has provided an incorrect explanation, leading to the recollection's defective behavior.

In the cup-and-pail example, the top set consist of the cup definition along with the brick, glass, bowl, and briefcase precedents. The bottom set consist of all those precedents in which the true-success suspicious relation, *handle is fixed,* causes something.

For our cup-and-pail example, our database contained about 100 miscellaneous precedents, including the brick, glass, bowl, and briefcase precedents, the cup definition, and two other precedents that turn out to be relevant, namely the door and straw precedents.

The *handle is fixed* relation appears in our database in only the door precedent, in which it causes *door is orientable.* Thus the bottom set consists of the door precedent alone.

Accordingly, when our program reexamines the *object is cup* relation, it uses the brick, glass, bowl, briefcase, and door precedents. It does not use the cup definition because the *object is cup* relation is caused by something in the cup definition. Our program fails because the top and bottom sets do not connect the reexamined relation, *object is cup,* to the suspicious relation *handle is fixed.*

Similarly, when our program reexamines the *object is stable* relation, it uses the cup definition along with the glass, bowl, briefcase, and door precedents, but fails again. When it reexamines the *object enables drinking* relation, it uses the cup definition along with the brick, bowl, briefcase, and door precedents. Again it fails, because it cannot connect *object enables drinking* to *handle is fixed,* as shown in figure 6.

Our program also fails when it goes on to try reexamining the *object transports liquids* and the *object is liftable* relations. Evidently, more precedents have to be used.

Incorporating true-success suspicious relations may require search

Once our program concludes that more precedents have to be considered, it augments the precedents in either the top or the bottom sets.

To augment the top set, our program identifies new precedents in which there is a relation with two properties: the relation must cause something in an existing top set precedent; and the relation must be

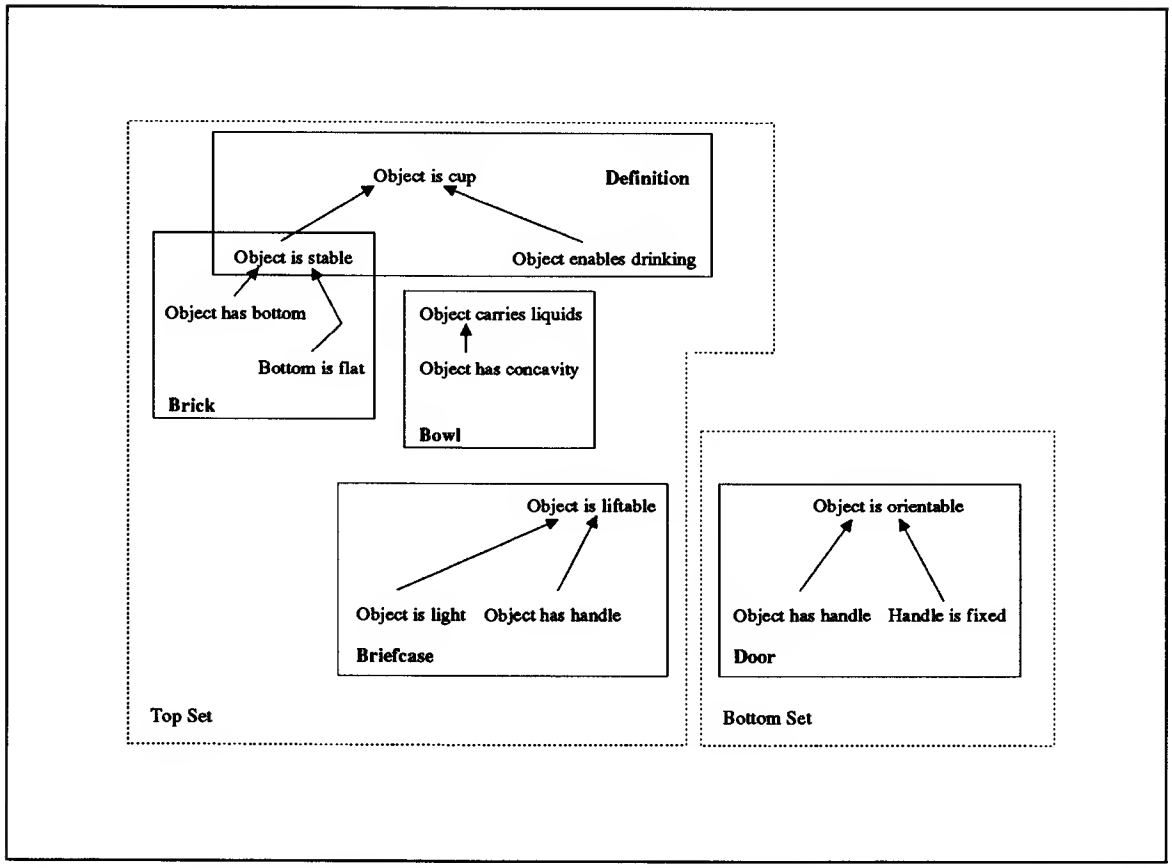

Figure 6. Reexamination fails for the top and bottom sets are not connected to one another. Only the relevant parts of the precedents are shown.

caused by something in the new precedent. Thus the new top-set precedents extend the *cause* chains that lead up through the existing top-set precedents.

Symmetrically, to augment the bottom set, our program identifies new precedents in which there is a relation with two properties: the relation must be explained, in part, by something in an existing bottom set precedent; and the relation must help to explain something in the new precedent. Thus the new bottom-set precedents extend the *cause* relation chains that start down in the existing bottom-set precedents.

To keep the number of precedents as limited as possible, our program augments the set with fewer precedents. In our example, this means augmenting the bottom set because it currently has only one precedent, the door precedent. The straw precedent is added because it contains the relation *straw is orientable*: in the straw precedent, *straw is orientable* causes *straw is manipulable*, and in the door precedent, *handle is fixed* causes *door is orientable*.

Now the reexamination process starts over with the augmented bottom set. As before, reexamination fails on the topmost relation, *object is cup*, and on the first of the relations in the next layer, *object is stable*. However, when our program reexamines the *object enables drinking* relation, it succeeds in connecting *object enables drinking* with the *handle is fixed* relation via the definition, the new straw precedent, and the existing door precedent, as shown in figure 7. Of course, all of the precedents shown in the figure contain details that are not shown so as to avoid clutter.

Note that the recollection's AND tree is restructured as necessary without reasoning explicitly about wrong or missing nodes and branches. The restructured AND tree can be viewed as the following if-then rule:

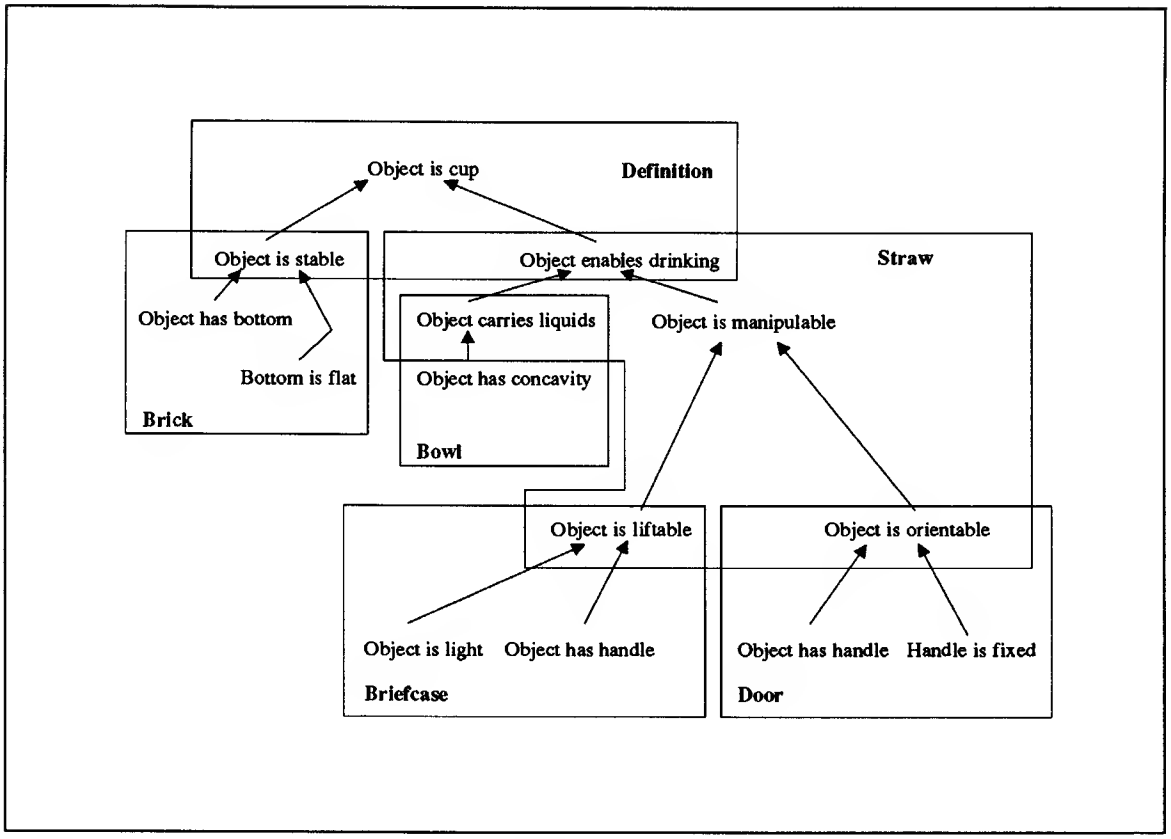

Figure 7. A recollection tree after repair with contributing precedents is shown. Note that the straw precedent, having augmented the bottom set, bridges the gap between the old recollection and the suspicious relation, *handle is fixed.* Now only cups are recognized as cups; pails are not. Only the relevant parts of the precedents are shown.

Repaired Cup-recognition rule:

If	The object has a bottom
	The bottom is flat
	An object has a concavity
	The object is light-weight
	The object has a handle
	The handle is fixed
Then	The object is a cup
Providing	The object is stable
	The object enables drinking
	The object carries liquids
	The object is manipulable

Our program tries to explain all true successes

Because we want to incorporate all of the true-success suspicious relations into the revised recollection tree, our breadth-first reexamination procedure works top-down. That way, later repairs cannot conflict with the contribution of earlier ones. The reexamination process stops when all true-success suspicious relations are incorporated into the revised recollection tree.

The Solution May Be To Explain the False-Success Suspicious Relations, Creating a Censor

The repaired recollection tree works on all of our cup-versus-pail problems because none of the pail descriptions contains a *handle is fixed* relation. There remains a danger, however, that our problem solver will try hard to show that a given pail has a fixed handle even though it already knows that the pail's description contains a *handle is hinged* relation.

Fortunately, however, our system not only repairs recollection trees but also builds new ones that can serve as censors. In the cup-and-pail example, the censor constructed, expressed as an *if-then* rule, is quite simple:

Hinged-handle Censor:

If Handle is hinged
Then Handle is not fixed

Once this censor is created, our system blocks any attempt to show that a handle is fixed, given that it is hinged. The recollection still recognizes the cups, and fails to recognize the pails, but now the censor blocks useless effort directed at showing that pails have fixed handles.

Censors are created after our repair program has finished explaining the true-success suspicious relations. To create censors, our program does a breadth-first reexamination of all the relations in the repaired recollection tree, looking for precedents that tie the negation of each reexamined relation to a false-success relation. The resulting explanations establish why the false-success suspicious relations should block recognition. These explanations, in turn, permit the creation of new censors.

Initially the precedent set is limited to the following to keep explanations as short as possible:

- Precedents in which the negation of the reexamined relation is caused by something. These precedents constitute the initial *top set.*
- Those precedents in which one of the false-success suspicious relations causes something. These precedents constitute the initial *bottom set.*

Here the idea is to find an explanation for the negation of the reexamined relation that includes at least one of the false-success suspicious relations along with some or all of the relations that lie in the intersection of all the successes. If our program finds such a collection of precedents, it creates a new censor from that collection of precedents.

In the cup-and-pail example, our program reexamines the *object is cup* relation, looking for a chain of precedents that link *object is not a cup* to the suspicious relation, *handle is hinged.* This reexamination must fail because there are no precedents in our database that show that something is not a cup.

Eventually, however, our program's breadth-first reexamination tries to show the *handle is not fixed* relation, given the false-success suspicious relation, *handle is hinged.* At this point, our limited database provides only one precedent, the briefcase, which finds its way into both the top and bottom sets because *handle is hinged* is connected to *handle is not fixed* by a cause relation. This explains the presence of the false-success relation and generates the new censor.

In general, when all false-success suspicious relations are incorporated into the revised recollection tree, the reexamination process stops because everything is completely explained. While false-success suspicious relations remain, the breadth-first reexamination procedure continues.

There Is a Panoply of Heuristic Choices

The keys to our program's operation are the isolation of suspicious relations using near-miss groups and the use of those suspicious relations in reducing the search for an explanation-based repair. In implementing our program, many choices were decided by heuristic arguments, or lacking anything else, plain simplicity. Here are some examples:

- Our program could search for the simplest explanation for a suspicious relation according to, say, the number of precedents involved in the explanation, or the number of links in the causal chains contained in the precedents, or the total number of links in the causal chain leading from the recollection's root,

through the precedents, to the suspicious relation. For simplicity, we decided to have our program be content with the first explanation found.

- Our program could terminate when just one suspicious relation has been incorporated into the recollection AND tree or into a new censor. We decided to have our program keep on looking for explanations for the other suspicious relations, within resource limits, so as to learn as much as possible.
- Our program could use smaller initial top sets when trying to explain true-success suspicious relations. We decided to include all of the original precedents to retain as much of the old recollection's AND tree as possible.
- There are many possible give-up conditions. We have our program give up after two rounds of precedent-set expansion on the ground that the more precedents involved, the flimsier the argument.

Many Situations May Contribute to Recollection Repair and Censor Creation

Minimally, only two descriptions are required to learn something as a byproduct of analogical problem solving: an exercise and a precedent. In the cup-recollection repair, many descriptions are involved: there are five precedents held over from the creation of the original recollection, two true-success situations, two false-success situations, two new precedents used to revise the recollection, and one precedent used to create the censor. Evidently, considerable knowledge can be brought to bear, in general:

- Many precedents and previously-generated recollections can contribute to the creation of a new recollection.
- Many successes and many near misses may be involved in isolating the suspicious relations.
- Many precedents and previously-generated recollections can contribute to a repair or to the creation of a new censor.
- Many cycles of isolation, repair, and censor creation may be needed to correct all problems.

Failure Stimulates a Search for More Detailed Descriptions

Note that it is the job of a benign teacher to be sure that recognition errors can be traced to suspicious relations. If there are no suspicious relations, there are two ways to correct the situation:

- First, although the lack of suspicious relations indicates that there is no common explanation for failure, there may be just a few explanations, each of which is common to a subset of the true successes or the false successes. The problem is to partition situations into groups, inside each of which there is a consistent explanation for failure.
- Second, the lack of suspicious relations may indicate that the situations needs to be described at a finer grain, adding more detail, so that an explicit explanation emerges.

Of these two ways to correct the situation, the more interesting is the one that leads to more detailed description. Typically, AI programs must be supplied from the beginning with all they need to know. Here, however, failure to find an explanation could initiate a search for more information.

Boris Katz has suggested a way of narrowing that search by hypothesizing oversights. Suppose, for example, that we fail to mention that the metal cup in figure 4 has a fixed handle. Then nothing lies in the intersection set, but the cup examples exhibit many relations, including the *handle is fixed*, *object is white*, and *object made-of metal* relations, that are not found in any pail description. Accordingly, we can temporarily assume that these relations are true-success relations, one at a time, looking for a relation that would enable our program to repair the recollection. Once such a relation is found, we can then ask the teacher to affirm that the relation holds in all the true-success situations and holds in none of the false-success situations. In our example, the question posed would be something like "I think the metal cup's handle must be fixed. Is it?"

Generalizing the oversight-hypothesizing approach, if the *handle is fixed* relation appears in some cups, but in none of the ones that happen to be involved in the near miss group, it still makes sense to test them with our procedure rather than asking about random relations.

Near-Miss Groups Solve an Old Problem

In 1970, Winston showed how the notion of what an arch looks like can be learned from samples and near misses. At that time, it was necessary to rank the importance of various relations *a priori* because it was rare that only one relation would emerge when comparing the current arch model with a near miss. In figure 8, the near miss can be explained in more than one way. The ambiguity can be resolved by supplying a second near miss and treating the two near misses as a near-miss group. Figure 9 illustrates this.

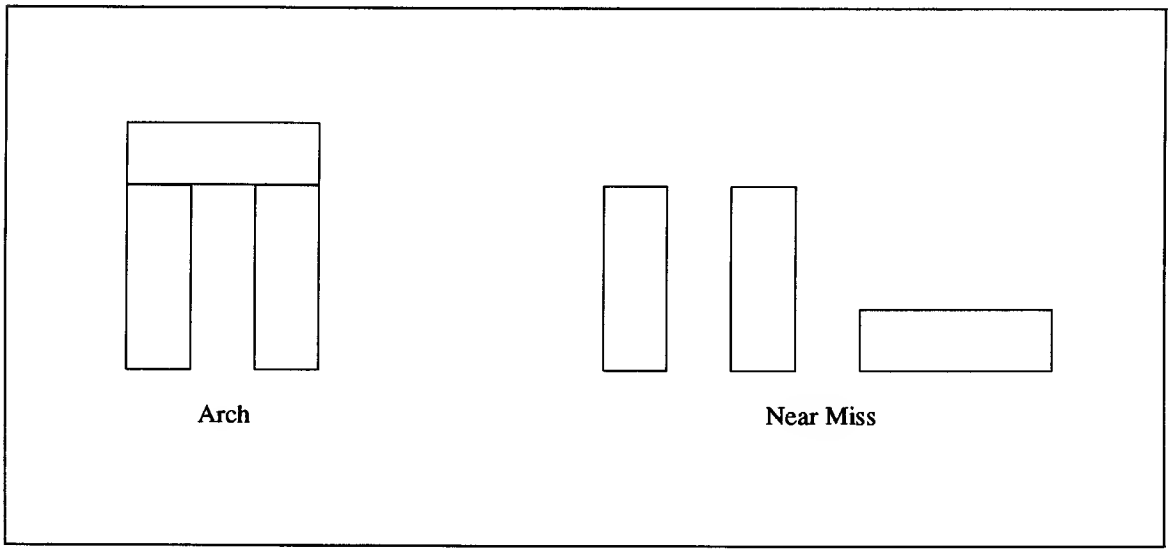

Figure 8. An arch and a near miss. One explanation is that the near miss is not an arch because it loses the *supported-by* relations between the lintel and the posts; another is that it is not an arch because it gains a *right-of* relation between the lintel and both of the posts.

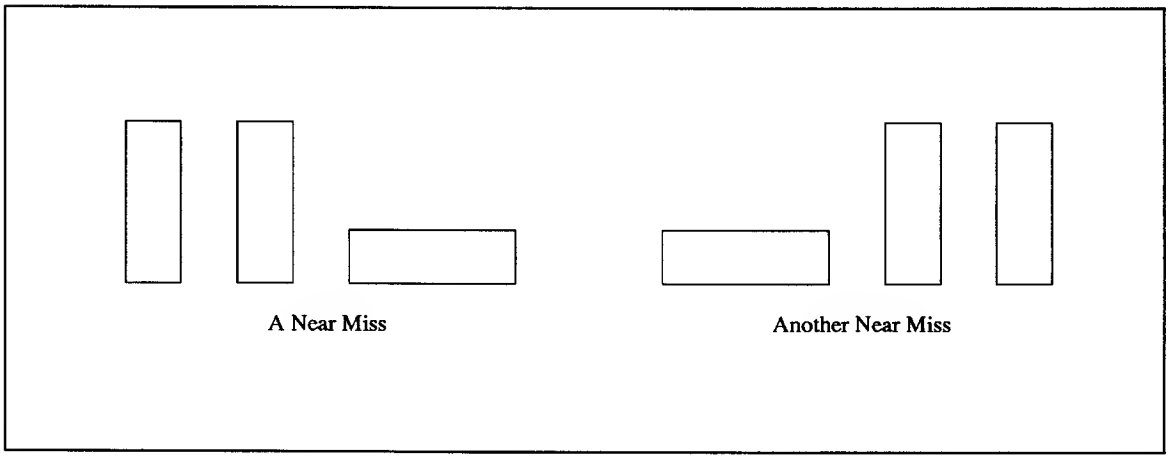

Figure 9. A near-miss group. The only common differences perceived between each of these and an arch is the lack of *supported-by* relations between the lintel and the posts.

Related Work

The work in this section is based mainly on early near-miss oriented learning work [Winston 1970] and on more recent work on extracting rules from Shakespearean plots [Winston 1980, 1982] and on deducing

what cups look like from precedents [Winston et al. 1983]. All of this is described in *Artificial Intelligence, Second Edition* [Winston 1984].

The idea of using differences to focus learning resonates with the difference-based reasoning work of Falkenhainer [1988] in which he uses differences between working devices and nonworking devices, plus domain knowledge, to deduce why the nonworking devices fail.

One defect of the idea described in this paper is that every concept has to have a name, but in many of our experiments, the concepts do not correspond well to English words, forcing us to invent awkward, multiply-hyphenated names. An approach to dealing with this defect is explained in a forthcoming thesis by Rao.

References

Falkenhainer, Brian [1988], "The Utility of Difference-Based Reasoning," *National Conference on Artificial Intelligence*, Saint Paul, Minnesota.

Kratkiewicz, Kendra [1984], "Improving Learned Rules Using Near Miss Groups," B.S. Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, with Winston, Ms. Kratkiewicz showed that near-miss groups could pull informative differences out of complicated descriptions of historic conflicts.

Mitchell, T. M., R. M. Keller, and S. T. Kedar-Cabelli [1986], "Explanation-based Generalization: A Unifying View," *Machine Learning*, vol. 1, no. 1.

Winston, Patrick Henry [1970], "Learning Structural Descriptions from Examples," Ph.D. Thesis, Massachusetts Institute of Technology. The arch example is introduced, along with the notion of near misses. A shortened version is in *The Psychology of Computer Vision*, edited by Patrick Henry Winston, McGraw-Hill Book Company, New York.

Winston, Patrick Henry [1980], "Learning and Reasoning by Analogy," *Communications of the Association for Computing Machinery*, vol. 23, no. 12.

Winston, Patrick Henry [1982], "Learning New Principles from Precedents and Exercises," *Artificial Intelligence Journal*, vol. 19, no. 3.

Winston, Patrick Henry [1984], *Artificial Intelligence, Second Edition*, Addison-Wesley. Analogical reasoning using precedents is described in detail Chapter 12. Ignore the description of the matcher—there are much better matchers now.

Winston, Patrick Henry, Thomas O. Binford, Boris Katz, and Michael R. Lowry [1983], "Learning Physical Descriptions from Functional Definitions, Examples, and Precedents," *National Conference on Artificial Intelligence*, Washington, DC. Also described in *Artificial Intelligence, Second Edition*, Addison-Wesley, by Patrick Henry Winston, 1984.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AIM 1231	2. GOVT ACCESSION NO.: A228711
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): Repairing Learned Knowledge Using Experience	5. TYPE OF REPORT & PERIOD COVERED: memorandum
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Patrick H. Winston and Satyajit Rao	8. CONTRACT OR GRANT NUMBER(s): N00014-85-K-0124
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Artificial Intelligence Laboratory 545 Technology Square Cambridge, MA 02139	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: Advanced Research Projects Agency 1400 Wilson Blvd. Arlington, VA 22209	12. REPORT DATE: May 1990 13. NUMBER OF PAGES: 14
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office): Office of Naval Research Information Systems Arlington, VA 22217	15. SECURITY CLASS. (of this report): UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

learning
knowledge repair
near-miss groups
precedent-based learning
explanation-based learning

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Explanation-based learning occurs when something useful is retained from an explanation exercise, usually an explanation of how some particular problem can be solved. If explanation is based on sound theory, then the learning process speeds up future problem solving, but the scope of the learning-augmented theory remains unchanged. In con-

(continued on back)

Block 20 continued:

trast, if explanation is based on fragmentary and often defective experience, explanation can be a guide to when that experience can be deployed. Thus one kind of explanation provides speed up; another kind of explanation provides new knowledge. Experience is not sound theory, however, and wrong things may be learned accidentally, as subsequent failures will likely demonstrate. In this paper, we describe ways to isolate the facts that cause failures, ways to explain why those facts cause problems, and ways to repair learning mistakes. In particular, our program learns to distinguish pails from cups after learned knowledge about cups leads a recognition program to think pails are cups.

CS-TR Scanning Project
Document Control Form

Date : 10/27/94

Report # AIM-1231

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☐ Technical Report (TR) ☒ Technical Memo (TM)
☐ Other:_____

Document Information

Number of pages: 14
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
☒ Single-sided or
☐ Double-sided

Intended to be printed as :
☐ Single-sided or
☒ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☒ Laser Print
☐ InkJet Printer ☐ Unknown ☐ Other:_____

Check each if included with document:

☒ DOD Form ORG ☐ Funding Agent Form ☐ Cover Page
☐ Spine 1(PGS) ☐ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages(by page number):_____

Photographs/Tonal Material (by page number):_____

Other (note description/page number):

Description : Page Number:

_____
_____
_____
_____

Scanning Agent Signoff:

Date Received: 10/27/94 Date Scanned: 10/27/94 Date Returned: 11/03/94

Scanning Agent Signature: Michael W. Cooper